John von Neumann Institute for Computing

Real-Time Simulation for Laser-Tissue Interaction Model

L.F. Romero, O. Trelles, M.A. Trelles

published in

Parallel Computing:
Current & Future Issues of High-End Computing,
Proceedings of the International Conference ParCo 2005,
G.R. Joubert, W.E. Nagel, F.J. Peters, O. Plata, P. Tirado, E. Zapata (Editors),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. **33**, ISBN 3-00-017352-8, pp. 415-422, 2006.

http://www.fz-juelich.de/nic-series/volume33

Real-time simulation for laser-tissue interaction model

L.F. Romero[a], O. Trelles[a], M.A. Trelles[b]

[a]Dept. Computer Architecture, University of Malaga, Spain

[b]Medical Institute of Vilafortuny, Cambrils, Tarragona, Spain

The extensive use of laser as a medical and surgical tool has lead to a growing interest in modelling the interactions between laser irradiation and human tissues. This modelling has enormous computational needs where the use of parallel computing can provide CPU-power enough to obtain results in real time, and with a proper representation. To this end we have developed a three-layer, parallel architecture in order to simulate the laser-tissue interaction. The first layer is used to obtain the irradiance distribution by Monte Carlo simulation under an optical model; Then, in the second layer, the temperature changes produced by the energy delivered by laser device is obtained by means of a differential equation based model. Finally, the thermal damage is predicted from the spatial and temporal temperature distributions. To achieve high efficiencies and real time results, the complexity of the model requires a complex parallel implementation. In this work, an interface for a hybrid parallel communication model is presented. This interface makes easier the programming of high efficiency hybrid codes, even with a reduced set of processors.

1. Introduction

Laser treatment based on controlled tissue elimination using selective photothermolysis is now well established as the treatment of first choice for various skin lesions and especially for the treatment of pigment disorders and skin tumors. When choosing a proper set of irradiation parameters (wave-length, pulse length, beam size, etc.) for a pulsing laser beam applied to a given target zone, some undesired kinds of tissue can be destroyed by inducing thermal damage in it, while the temperature of the surrounding tissues is kept below the threshold for damage.

However the optimal choice of laser irradiation parameters and guidance of treatment is closely related to the prognosis of results. This is a very complex problem because it is strongly associated with the specific lesion characteristics with an enormous variety in histopathology of skin disorders, the surrounding tissue and its particular structure distribution in the various skin layers, the type of laser device with a diversity of laser irradiation parameters, among others. The many issues involved in this problem made of it a field of actual interest that claims for an in-deep research. In the last, this problem has lead a growing interest in modelling the interaction between laser irradiation and human tissues.

We have address this problem by a new laser/tissue interaction model based on three different layers: (a) First, the irradiance distribution –how light delivered by laser device propagates through such tissue models– is determined by Monte Carlo simulation; next (b) the temperature distribution in the tissue caused by laser energy deposition is estimated by solving the *bioheat* transfer equations; and lastly (c) the thermal damage is predicted from the spatial and temporal temperature distributions, with the aid of so-called damage integral Arrhenius formulation, in which the thermal damage to tissue is described as a temperature dependent rate process. The three layers architecture, presented in Section 2 allows a close reproduction of the effect of laser with a realistic tissue model. Figure 1 shows a schematic representation of the model, on the left, and the most important dependencies among the layers and its parameters, on the right.

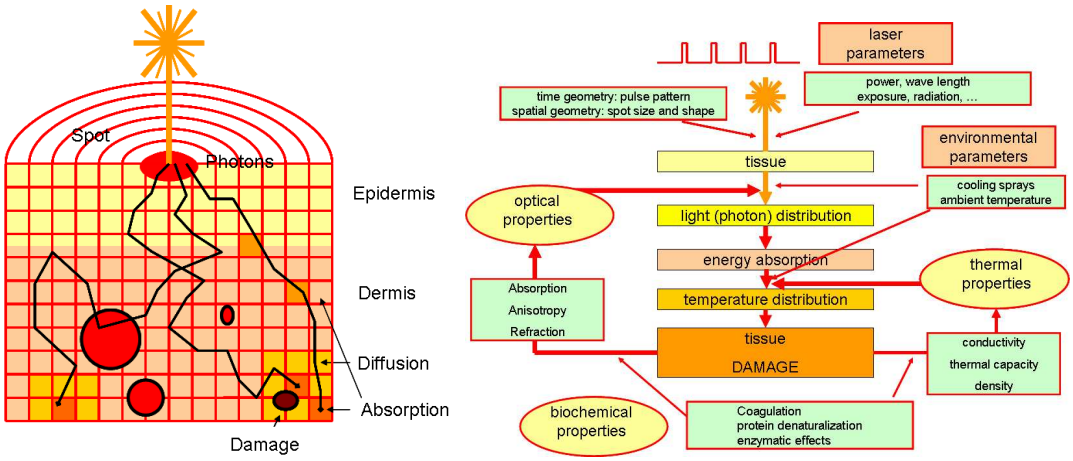

Figure 1. A laser tissue interaction scheme and a dependency diagram of the model.

Parallel computing with low-cost multiprocessor systems have been employed to achieve the real time simulations required for monitoring the laser-tissue interaction. While the Monte Carlo simulation in parallel is trivially solved, the major computational cost and complexity arise from the integration of the equation describing the heat transfer. To obtain a faster computation with good accuracy, a finite differences method in parallel is employed here, using a block distribution strategy for the discretization grid. Finally, Arrhenius formulation takes profit of the grid distribution and no data interchange is necessary in this step. The parallelization of the model is presented in Section 3.

One of the major advantages using non-deterministic techniques in the simulation process lays in the fact that processors synchronization is not strictly necessary to obtain information. Different parallelization techniques and communication paradigms have been combined to optimize response time producing high efficiency and near real time results. In Section 4 an interface to make easier the programming of efficient hybrid codes is presented.

The final integrated application allows to conduct the full simulation in real time (shown in Section 5) after the configuration parameters are settled. A multi-layer description of tissue allows a very close representation of the different tissue irregularities, with a easy definition of the diverse tissue components such as chromospheres, small veins, dermis, epidermis, etc, and even it is possible to incorporate external sources like as cryogen gels or to modify the device parameters.

2. The model

The three layers of the computational scheme of the laser/tissue interaction model are strongly related. As stated above, the photon energy of a pulsing laser in layer 1 increases the temperature in layer 2, and the induced heat modifies the cells in layer 3. As the thermal and optical properties are modified in a damaged tissue, there is a feedback of layer 3 over the first two layers of the proposed model. In order to obtain a realistic model, an time iterative implementation of the three layers is required, in which the time step should be as short as possible. But in practice, the cell degradation by effect of the temperature, and so, the changes of the optical and thermal properties, are governed by a time factor which is much larger than the required precision in the solution of the diffusion equation for typical grid sizes. For this reason, a compromise solution with a relaxed interaction

(with two levels of time steps) among the layers have been considered, in order to minimize the communication overhead of the model and to increase its temporal locality:

```
Algorithm 1:      for i = 1,..., number of global time steps
                     compute photon/energy distribution (layer 1)
                     for j = 1,..., number of diffusion time steps
                          Energy absorption=f(energy distribution,pulsing laser time shape)
                          compute thermal diffusion (layer 2)
                     compute damage (layer 3)
```

2.1. Light distribution

Light distribution is estimated by launching photons governed by the optical properties of tissue (scattering, absorption, etc) [3]. The number of photons must be large enough to obtain statistically validated results. In the travel through the tissue, every photon deploys some of its energy in small cells (*voxels*) resulting from a discretization of the tissue. The optical properties of each *voxel* depends on the tissue layer to which it belongs [1]. The process is represented by this algorithm:

```
Algorithm 2:      for i = 1,..., number of photons
                     launch photon
                     while (photon in grid) and (photon alive)
                          move photon
                          compute scattering
                          if (layer boundary)compute refraction and reflection
                          compute energy deposition
                          if (energy < threshold) photon not alive
```

The Monte Carlo method employed here requires a high computational power for statistically valid simulations, but more accurate results can be obtained in comparison with other methods.

2.2. Thermal difusion

The heat transfer is modelled in this work by the *bioheat* equations with advection (1), and the corresponding contour conditions [5,6]:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho \cdot C}(D \cdot \frac{\partial^2 T}{\partial x^2} + D \cdot \frac{\partial^2 T}{\partial y^2} + \frac{\partial}{\partial z} D \cdot \frac{\partial T}{\partial z} + F_{met} + F_{circ} + A + E_{laser}, \quad (1)$$

where T is the temperature, ρ is the density, C is the specific heat, k is the thermal conductivity, D is the diffusion coefficient; F_{met}, a heat source from the cellular metabolism; F_{circ}, a heat source from the smaller blood vessels; A is the advection (see Figure 2), and finally, E_{laser} is the energy carried by the photons, computed in the previous layer. Due to the typical size of photon sampling and the dimensions of the integration grid for the partial differential equation system, our choice has been pre-computing the energy distribution matrix based on light distribution in the previous step and on the time shape of the pulsing laser.

Equation(1) has been integrated using a Crank–Nicolson finite difference method. It is noteworthy to observe that both spatial and temporal discretization in this step are the same as used in the Monte Carlo simulation. Once the energy distribution is known and the matrix coefficients have

[1] A typical simulation uses $50 \times 50 \times 50$ *voxels*, each one of size $100\mu m \times 100\mu m \times 100\mu m$.

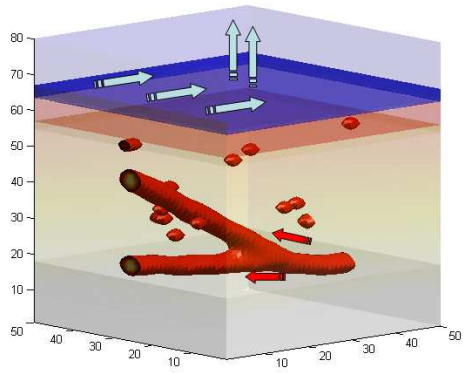

Figure 2. A complex model with an air layer, cryogen gel, epidermis, dermis, subcutaneous tissue, vessels, chromospheres, and several advection terms (represented with arrows).

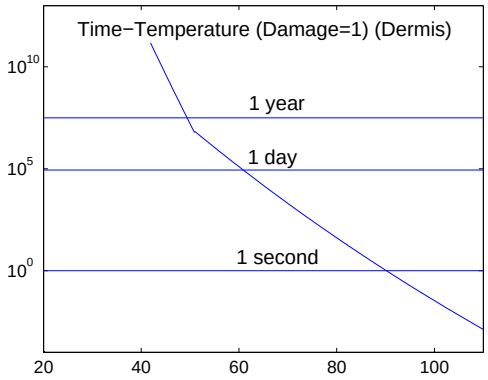

Figure 3. Exposure time required to damage a 100% of dermis cells, as a function of temperature.

been computed, the resulting system of $n = n_x \times n_y \times n_z$ linear equations is solved using the Preconditioned Conjugate Gradient (PCG) method.

2.3. Arrhenius formulation for the tissue damage

The thermal increase modifies the physiological properties of a tissue. Normally, when temperature rises to about 50°C, there is a protein denaturalization which induces the death of cells in short time. Arrhenius formulation [1] is employed here to calculate the accumulative damage ($\Delta\Omega$), which is irreversible when the total induced damage affect to the 100% of cells (Ω=1). Arrhenius equation (2) computes the accumulative damage in a tissue, exposed to a given temperature for a certain time:

$$\Delta\Omega(T,t) = A\int_{t_i}^{t_f} e^{-\frac{E_a}{R\cdot T}}dt \simeq A(t_f - t_i)\cdot e^{-\frac{E_a}{R\cdot T}}, \tag{2}$$

where A is the frequency factor, $t_f - t_i$ is the exposure time, R is the universal gas constant, and E_a is the energy activation barrier. In Figure 3 the time required to an irreversibe damage of dermis tissue is shown for different temperatures. Usually, the damage will be mainly produced in the epidermis (due to its proximity to the laser source), and also in the haemoglobin (because of its high absorption coefficient). The use of cryogen gels (below 0°C) will minimize the damage in the epidermis [2].

3. Parallel implementation

In order to achieve real time realistic simulations, a parallel implementation has been developed, which can be properly scaled in order to get the maximum performance from the available resources. Up to four level of parallelism have been combined in a typical implementation of the model, as described below (Fig. 4):

1) A client-server model, usually using a PC acting as front-end client, sending to a simulator Server the control parameter of the model (such as intensity, pulse shape, environmental conditions, etc.). The front-end client also renders any volume data which have been received from the simulator Server. The experiment presented below have been obtained using a PC with a Pentium 4 as client, and an Altix 3000 as server.

2) The Altix server launches two MPI processes. The first one computes the trajectory of photons (the *photon server*), while the other one solves the bioheat equation (the *grid server*).

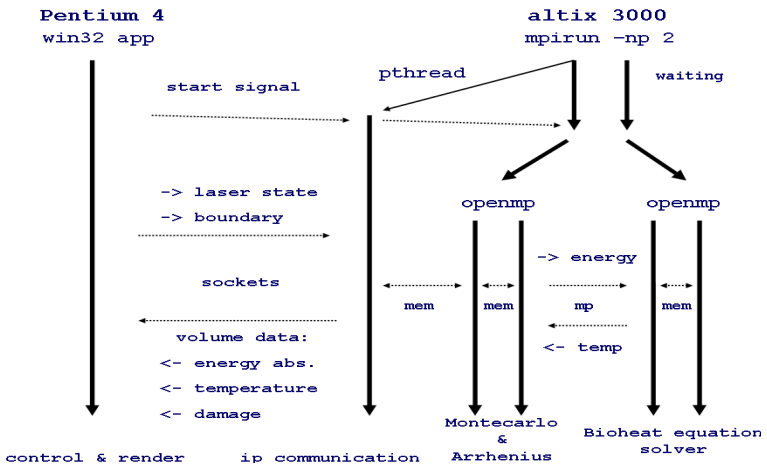

Figure 4. Gantt chart for a typical parallel implementation

3) Both MPI processes have been parallelized using a shared memory model with OpenMP. In the first MPI process, Monte Carlo is parallelised distributing the number of photons among the different processors and collecting results in a master thread which will communicate with the grid server. Photon trajectories are independent allowing an efficient parallelization. In the second one, a block distribution strategy for the discretization grid following the z-direction is used, in such a way that each processor computes the matrix coefficients and the right hand sides in their assigned nodes. This is equivalent to a block distribution of the system matrix if a natural ordering is used. The resulting system is solved by means of the PCG method, in which each vectorial operation is solved in parallel by a block distribution of the vector components among processors.

4) In addition, one of the MPI processes (usually the *photon server*) launches a Posix thread, which will perform an asynchronous communication with the PC client.

3.1. Communication pattern

Note that the communication between the processes and threads may be synchronous or asynchronous, depending of the requirements in each level. It has to be taken into account that, like reality, this is an stochastic model, in which the computational rigourousness in the solution of the problem can be partially sacrificed without a penalty in the realism of the results. So, the PC client and the server machine only requires asynchronous communication: the posix thread will continuously send the data as they are found in the memory, in any moment, and it will forward the control commands received from the PC to a memory location in one of the MPI processes. The volume data for visualization is sent by the server using just one byte per voxel in a compressed package. Decompression and rendering of the volume data is performed by an event driven application using the fast VTK visualization library [7]. The communication rate only depends on the LAN connection and the frequency of this asynchronous communication is large enough to avoid a visual mismatch.

The *photon server* and the *grid server* also communicates each other by using an asynchronous message passing model (MPI_Isend, MPI_Irecv) at a frequency of about 10Hz. This frequency is large enough to eliminate numerical instability and to provide realistic simulations. Note that the protein denaturalization, which modifies the optical parameters for a temperature increase, usually occurs at a rate of tens of seconds. On the other hand, communications in the *grid server*, required for the solution of equation (1), which occurs through the memory, must be carefully synchronized.

3.2. Load balancing

An efficient parallelization strategy for the PCG solver has been used in this work, which will ensure a minimization of the communication cost and a good load balancing inside the *grid server* [4]. As the integration of equation (1) establishes the temporal reference for the system, the computational load of the *photon server* can be easily adjusted (with a minimum threshold, statistically stablished), depending on the employed number of threads.

4. A communication model for hybrid systems

In this section, an interface to make easier the hybrid programming of such a complex parallel model is presented. The proposed idea is to make an abstraction of both the data interchange and the synchronization requirements, by solving the underlying problem in any parallel programming paradigm: the read-after-write (RAW) and write-after-read (WAR) hazards for any block of data being shared by two or more processors. Note that all parallel programming environment existing in the literature shares the information by using one of the following strategies: a) storing information in a shared memory location; b) copying remote information in a local storage: *get*; c) copying local information in a remote storage: *put*; and finally, by using a *send-receive* symmetric communicator.

The interface proposed here is based in the substitution of the mentioned operation (*put, get, send, recv*) and of any required syncronization (flags, semaphores, locks, etc.), by the use of only four routines, which has to be carefully inserted after and before the use of shared data. These routines may use any of the traditional strategies inside, depending of a the requirements, as explained below:

- *pre-write*: Routine used to evaluate a WAR hazard just before any modification of local data which is being shared. This call is only required in a shared memory environment, or when a *get* communicator is employed. Usually, it consist in evaluate if a certain *flag* variable,described below, is zero (WAR check).

- *post-write*: This routine is used after modifying the data, and includes three stages. Firstly, WAR hazard is evaluated, only if a *put* communicator is employed. It evaluates the availability of a remote location. Second, the data is delivered if required, by using *send* or *put*. And third, a *flag* variable is lifted, as the initial step of a RAW hazard detection, if a symmetric communicator is not being used (RAW release).

- *pre-read* : Routine used to receive data before any computation using it. It also includes three stages. Firstly, the *flag* variable is tested (if a symmetric communicator is not being used), as the final step of a RAW hazard detection. Second the data is received, by using *recv* or *get*. And third, the *flag* variable is done zero, as the liberation phase of a WAR hazard detection. The third stage is only required when a *get* communicator is employed. (RAW check).

- *post-read*: Routine used to liberate a WAR hazard just after using data produced in another processor. The *flag* variable is done zero. This call is only required in a shared memory environment, or when a *put* communicator is employed (WAR release).

In Table 1, the four operations of the proposed interface are summarized. Note that these routines should be applied to large blocks of data, rather than to individual variables (in the same way that messages should be unified, if possible, in a message passing model). For example, a *post-write* call should be invoked just after the last local modification, before an access from a remote processor. In general, the routines presented here should be placed immediately after or before the read or write operations, to ensure a minimization of the waits, and to overlap communications and computations.

The synchronization flags described here can be reused by different communication operations with some care. For example, in a global reduction, all flags involved can be replaced by a single counter. Also, if a same operation is performed inside a loop, an even-odd pair of flags should be alternatively used, to prevent from hazards affecting to the flag variable. Finally, when any of the paired routines (either a *post-read/pre-write* for WAR hazards, or a *post-write/pre-read*, for a RAW hazard) are dynamically separated by another pair, involving, at least, at the same PEs, then the synchronization operations over the flags in the first pair can be eliminated, because the hazard has been solved by the second one. So, many of the WAR hazard detections can be eliminated.

By considering all this items together, the resulting code should minimize the communication cost, and it will significatively reduce the waits in the synchronization points due to load unbalance.

Communicator	pre–write	WR	post-write	pre-read	RE	post–read
shared	test flag==0;		flag=1;	test flag==1;		flag=0;
symm MP			send;	recv;		
put			test flag==0; put; flag=1;	eval if 1		flag=0;
get	test flag==0;		flag=1;	test flag==1; get; flag=0;		

Table 1

PEs	task distribution	Ex. time(grid server)	Launched photons	speedup MC	speedup ED
1	1 (photon server)	-	13050	1	-
2	1 (photon server), 1 (grid server)	4.88 sec.	63650	-	1
4	1 (photon server), 3 (grid server)	0.98 sec.	12950	0.99	4.98
8	2 (photon server), 6 (grid server)	0.50 sec.	26550	2.02	9.76

Table 2

5. Results

In this section, we present some results, which has been obtained using an Altix 3000 system as the simulator server. Several experiments has been performed, using a typical skin model with 3 vessels, several chromospheres, and a discretization grid of 125000 voxels (see Figure 5). In all cases, times are shown for a 1 second simulation (10 global times steps, 10×20 diffusion time steps). The first experiment (using 1 PE as a photon server), has been used to determine how many photons can be launched by 1 PE in one CPU second. In the second and third experiments, one of the Itanium processor has been used as a photon server, while 1 and 3 (respectively) CPUs has been used for the solution of the bioheat equation (the CPU usage for the Arrhenius problem is irrelevant). In the third experiment, the parallel efficiency obtained (1.66) is very high, which is not only due to the cache effect, but also to a total absence of waits in the synchronization points. This is thanks to the use of the technique described in the previous section, which has allowed to eliminate all global barriers, and any unnecessary test for hazards. Real time results have been obtained in this case (note that many of the model properties have been previously adjusted to reach a real time simulation for this configuration of the system). The last experiment indicates that the scalability of the problem is very good for a larger number of processors, thus allowing a more complex modellization of the problem.

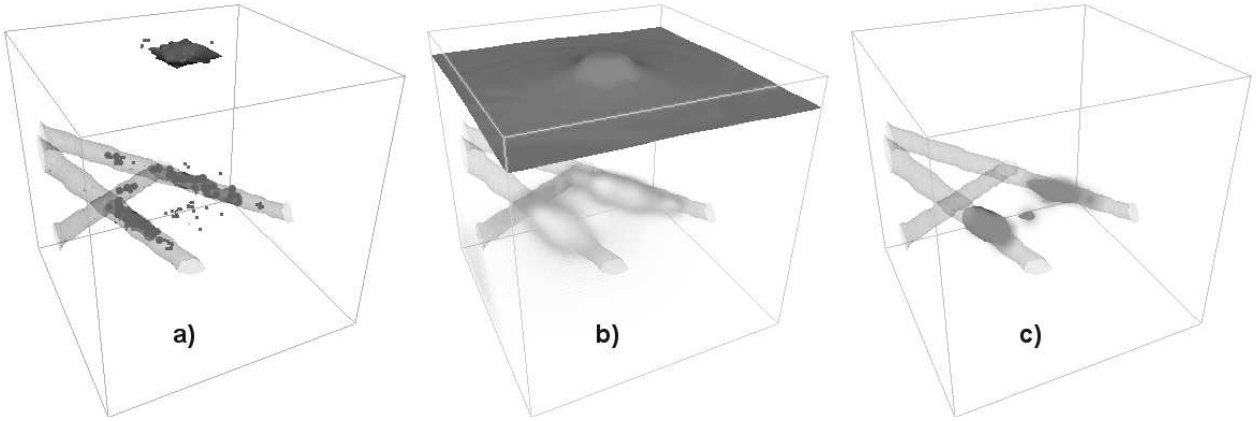

Figure 5. Simulation snapshots: a) photon energy deposition, b) tissue temperature, and c) induced damage for a sample model with three vessels. Figure b) includes a 0°C isosurface in the epidermis, and an opacity render (centered in 90°C) which is mainly located around the vessels.

6. Conclusions

In this work, an integrated model for the laser-tissue interaction has been presented. By means of the three-layer architecture of the model, an accurate representation of the tissue response to the laser stimulus can be achieved because of the extended set of parameters considered. Also, the three-layer architecture of the model incorporates a complex multi-level parallelism. Several communication paradigms and synchronization techniques have been included, because of the requirements of each level. To make easy the programming of the model, a new parallel programming interface for communications and synchronizations is proposed, which is independent of the parallelization paradigm.

With this interface, a robust, efficient and accurate parallel model has been obtained, which can achieve real time simulations of a complex model, even using a small parallel architecture. These results represent an step forward in assist greatly in diagnosis and treatment, and subsequently objectively assess the device parameters at various stages of treatment.

7. acknowledgement

The authors wish to thank Victor Espigares for his help in the integration of the model.

References

[1] J.K.Barton, A.Rollins, S.Yazdanfar, T.J.Pfefer, V. Westphal, J. A.Izatt, "Photothermal coagulation of blood vessels", *Physics in Medicine and Biology*, vol.46, pp.1665-1678, 2001.

[2] T.J. Pfefer, D.J. Smithies, T.E. Milner, M.J. Van Gemert, J.S. Nelson, A.J. Welch, "Bioheat transfer analysis of cryogen spray cooling during laser treatment of port wine stains", *Lasers in Surgery and Medicine*, vol.26, pp.145-157, 2000.

[3] A. Roggan, G. Müller, *Dosimetry and Computer Based Irradiation Planning for Laser-Induced Interstitial Thermotherapy*, SPIE Institute Series IS 13, Müller-Rogan Eds., 1995.

[4] L.F. Romero, E.M. Ortigosa, J.I. Ramos, "Parallel scheduling of the PCG method for banded matrices rising from FDM/FEM", *Journal of Parallel and Distributed Computing*,vol.63, pp.1243-1256, 2003.

[5] M.J. Van Gemert, A.J. Welch, J.W. Pickering, *Modelling Laser Treatment of Port Wine Stains*, O.T.Tan Eds., Amsterdam, 1992.

[6] M.J. Van Gemert, *Optical-Thermal Response of Laser-Irradiated Tissue*, Plenum, London, 1995.

[7] L. Avila *et al,* The VTK User's Guide*, Kitware Inc., 2004.*